

PPPLib User Manual

Author: [Chao Chen](#)

Update: September 12, 2020

CONTENTS

INTRODUCTION	3
FEATURES	3
HOW TO USE	4
LINUX USER	4
WINDOW USER	6
CONFIGURATION	8
DATASET	9
STATIC GNSS DATA	9
SUBURBAN VEHICLE DATA	9
URBAN DATA	9
PYTHON SCRIPTS	10
APPENDIX	16
PPP MODELS	16
RESULTS ANALYSIS FOR SINGLE-FREQUENCY	20
COMPARISON OF IONOSPHERE-FREE AND UNCOMBINED	22
EXAMPLE OF TRIPLE-FREQUENCY	24
ACKNOWLEDGEMENT	26
CONTACT AUTHOR	26
REFERENCES	26

Introduction

Precise Point Positioning Library (PPPLib) is a multi-GNSS data processing software designed to process multi-frequency data from GPS, BeiDou, Galileo, GLONASS and QZSS. PPPLib is written in the C/C++ programming language. It can compile, run on both Linux and Windows operating system. PPPLib mainly performs precise point positioning from single- to triple-frequency based on ionosphere-free or uncombined observations. Moreover, it provides abundant solutions including positioning, tropospheric delay, ionospheric delay as well as ambiguity information. Useful scripts and visualization tool are also provided for data download, batch processing or solution presentation. We give a preliminary review, including positioning accuracy and convergence time of PPP using dual-frequency, ionosphere-free from single-system to multi-GNSS, to show the working status of current version of the software. In addition, the software also supports post-processing kinematic mode and INS/GNSS loosely coupled mode for real kinematic scene.

Features

The main functions include:

- Supports multi-GNSS or their combinations with each other
- Standard single point positioning
- Single-frequency ionosphere-free PPP
- Dual-frequency ionosphere-free/uncombined PPP
- Triple-frequency ionosphere-free with one combination or two combination/uncombined PPP
- Support BDS-3 new satellites and signals
- Support PPP-AR with IRC or FCB (under testing)
- Multi-GNSS post-kinematic processing
- INS/GNSS loosely coupled (under testing)
- INS/GNSS tightly coupled (under testing)
- Convenient visualization

How to use

PPPLib uses CMAKE for project management. Currently the software can compile and run in Linux and Windows. We have tested the code under the virtual machine Ubuntu 16.0, Ubuntu 16.0 and Win10. PPPLib uses the [easyloggingpp](https://github.com/amrayn/easyloggingpp) library to log information and [Eigen](https://github.com/artsy/eigen) is used to perform matrix operations. And easyloggingpp is single header-only library for C++ applications. If there are any problems about easyloggingpp and Eigen library, please refer to the website <https://github.com/amrayn/easyloggingpp> and <https://github.com/artsy/eigen> for help. **Note that the author uses JetBrains CLion 2019.3 software platform for developing and debug under Ubuntu 16.0.** The CMAKE version is 3.15.3 and the C++ compiler is gcc 5.4.0. The personal configuration as shown in Fig. 1.

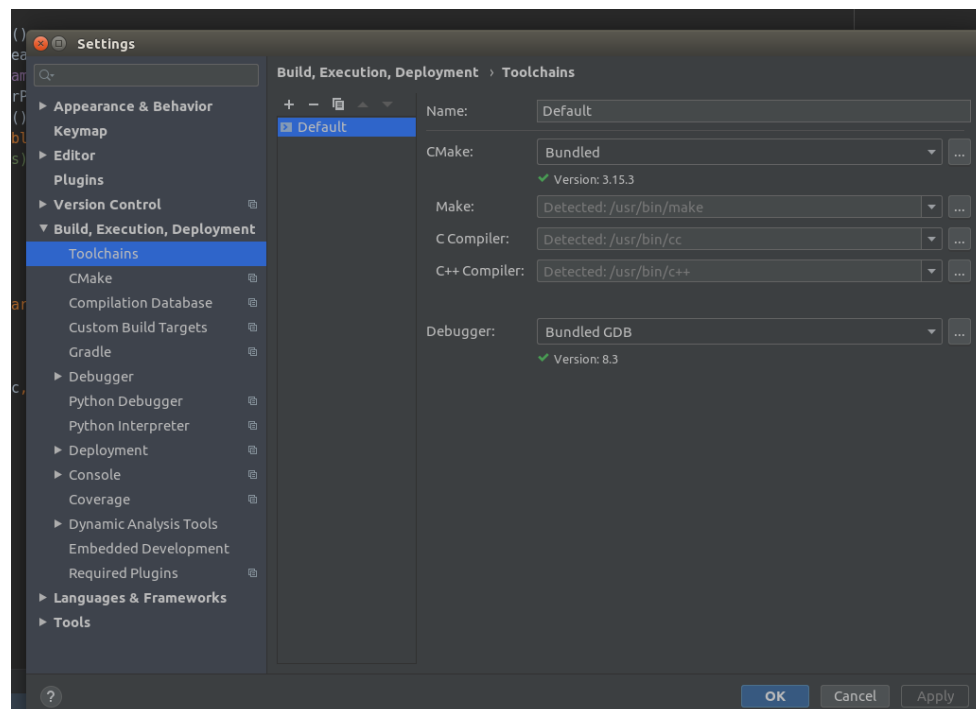


Fig. 1. Personal configuration on CLion platform under Ubuntu 16.0

Linux user

You can build and use the project follow steps 0-6:

0. Open Terminal, and cd [your path]
1. git clone <https://github.com/heiwa0519/PPPLib.git>
2. mkdir build and cd [your path]/PPPLib/build
3. cmake ..
4. make

After four steps, the executable files are generated in the [your path]/PPPLib/bin folder. You can use PPPLib in Terminal like:

5. cd ../bin
6. ./PPPLibMain -C ../conf/PPP/PPPLib_MGEX.ini -M PPP-KINE -S G -L 1

Note that the path in configuration file should set to your local directory.

1. clone and cmake

```
cc@hp: ~/Code/PPPLib/bin
File Edit View Search Terminal Help
cc@hp:~$ mkdir Code
cc@hp:~$ cd Code
cc@hp:~/Code$ git clone https://github.com/heiwa0519/PPPLib.git
Cloning into 'PPPLib'...
Username for 'https://github.com': heiwa0519
Password for 'https://heiwa0519@github.com':
remote: Enumerating objects: 471, done.
remote: Counting objects: 100% (471/471), done.
remote: Compressing objects: 100% (376/376), done.
remote: Total 471 (delta 77), reused 471 (delta 77), pack-reused 0
Receiving objects: 100% (471/471), 1.23 MiB | 6.00 KiB/s, done.
Resolving deltas: 100% (77/77), done.
Checking connectivity... done.
cc@hp:~/Code$ cd PPPLib/
cc@hp:~/Code/PPPLib$ mkdir build
cc@hp:~/Code/PPPLib$ cd build
cc@hp:~/Code/PPPLib/build$ cmake ..
-- The C compiler identification is GNU 5.4.0
-- The CXX compiler identification is GNU 5.4.0
-- Check for working C compiler: /usr/bin/cc
-- Check for working C compiler: /usr/bin/cc -- works
-- Detecting C compiler ABI info
-- Detecting C compiler ABI info - done
-- Detecting C compile features
-- Detecting C compile features - done
-- Check for working CXX compiler: /usr/bin/c++
-- Check for working CXX compiler: /usr/bin/c++ -- works
-- Detecting CXX compiler ABI info
-- Detecting CXX compiler ABI info - done
-- Detecting CXX compile features
-- Detecting CXX compile features - done
-- Found PythonLibs: /usr/lib/x86_64-linux-gnu/libpython2.7.so (found version "2.7.12")
-- Configuring done
-- Generating done
-- Build files have been written to: /home/cc/Code/PPPLib/build
```

2. make and build

```
cc@hp: ~/Code/PPPLib/bin
File Edit View Search Terminal Help
cc@hp:~/Code/PPPLib/build$ make
Scanning dependencies of target LogLib
[ 4%] Building CXX object src/LogInfo/CMakeFiles/LogLib.dir/LogInfo.cc.o
[ 8%] Building CXX object src/LogInfo/CMakeFiles/LogLib.dir/easylogging++.cc.o
[12%] Linking CXX static library ../../bin/libLogLib.a
[12%] Built target LogLib
Scanning dependencies of target CmnLib
[16%] Building CXX object src/CmnFunc/CMakeFiles/CmnLib.dir/CmnFunc.cc.o
[20%] Linking CXX static library ../../bin/libCmnLib.a
[20%] Built target CmnLib
Scanning dependencies of target OutLib
[24%] Building CXX object src/OutSol/CMakeFiles/OutLib.dir/OutSol.cc.o
[28%] Linking CXX static library ../../bin/libOutLib.a
[28%] Built target OutLib
Scanning dependencies of target InsLib
[32%] Building CXX object src/InsFunc/CMakeFiles/InsLib.dir/InsFunc.cc.o
[36%] Linking CXX static library ../../bin/libInsLib.a
[36%] Built target InsLib
Scanning dependencies of target GnssLib
[40%] Building CXX object src/GnssFunc/CMakeFiles/GnssLib.dir/GnssAR.cc.o
[44%] Building CXX object src/GnssFunc/CMakeFiles/GnssLib.dir/GnssErrorModel.cc.o
[48%] Building CXX object src/GnssFunc/CMakeFiles/GnssLib.dir/GnssFunc.cc.o
[52%] Linking CXX static library ../../bin/libGnssLib.a
[52%] Built target GnssLib
Scanning dependencies of target ReadLib
[56%] Building CXX object src/ReadFiles/CMakeFiles/ReadLib.dir/ReadFiles.cc.o
[60%] Linking CXX static library ../../bin/libReadLib.a
[60%] Built target ReadLib
Scanning dependencies of target DecodeRawLib
[64%] Building CXX object src/DecodeRaw/CMakeFiles/DecodeRawLib.dir/DecodeRaw.cc.o
[68%] Linking CXX static library ../../bin/libDecodeRawLib.a
[68%] Built target DecodeRawLib
Scanning dependencies of target SolverLib
[72%] Building CXX object src/Solver/CMakeFiles/SolverLib.dir/Solver.cc.o
[76%] Linking CXX static library ../../bin/libSolverLib.a
[76%] Built target SolverLib
Scanning dependencies of target AdjLib
[80%] Building CXX object src/AdjFunc/CMakeFiles/AdjLib.dir/AdjFunc.cc.o
[84%] Linking CXX static library ../../bin/libAdjLib.a
[84%] Built target AdjLib
Scanning dependencies of target PPPLib
[88%] Building CXX object src/CMakeFiles/PPPLib.dir/PPPLib.cc.o
[92%] Linking CXX static library ../../bin/libPPPLib.a
[92%] Built target PPPLib
Scanning dependencies of target PPPLibMain
[96%] Building CXX object exe/CMakeFiles/PPPLibMain.dir/PPPLibMain.cc.o
[100%] Linking CXX executable ../../bin/PPPLibMain
[100%] Built target PPPLibMain
```

3. run

```
cc@hp:~/Code/PPPLib/bin$ ./PPPLibMain -C ../conf/PPPLib_PPP.ini -M PPP-KINE -S G -L 128
[INFO] PPPLibMain.cc(303) | PROCESS DATA DIR: /home/cc/dataset/data_mgex
[INFO] PPPLibMain.cc(357) | === START PROCESS: harb
[INFO] Solver.cc(1045) | TOTAL EPOCH: 2880, SOLVE SUCCESS EPOCH: 2880, SOLVE FAILED EPOCH: 0
total(s): 14.3893cc@hp:~/Code/PPPLib/bin$
```

Window user

You should install the MinGW on your Windows computer for gcc compiler. It is recommended to use CLion IDE for run and debug. The personal configuration using Win10 as shown in Fig. 2. Load the PPPLib project, after cmake completed, you can add program arguments “-C ../conf/PPPLib.ini -M PPP-KINE -S G -L 128 -T 2019/12/01” in Run/Debug Configurations and then the project is ready for running.

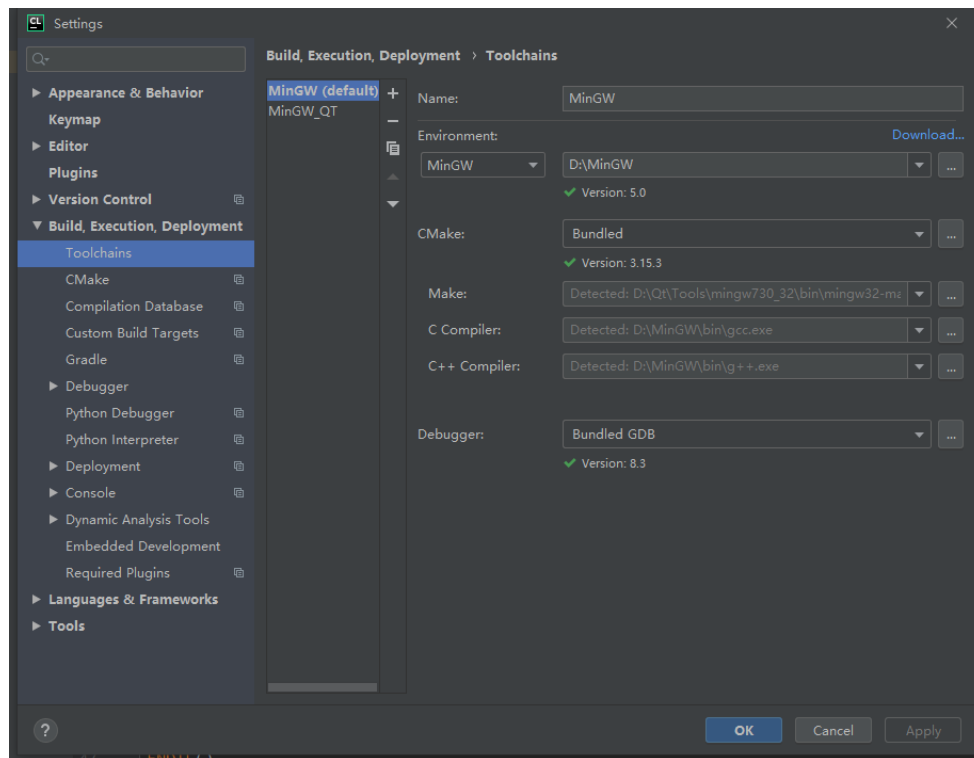
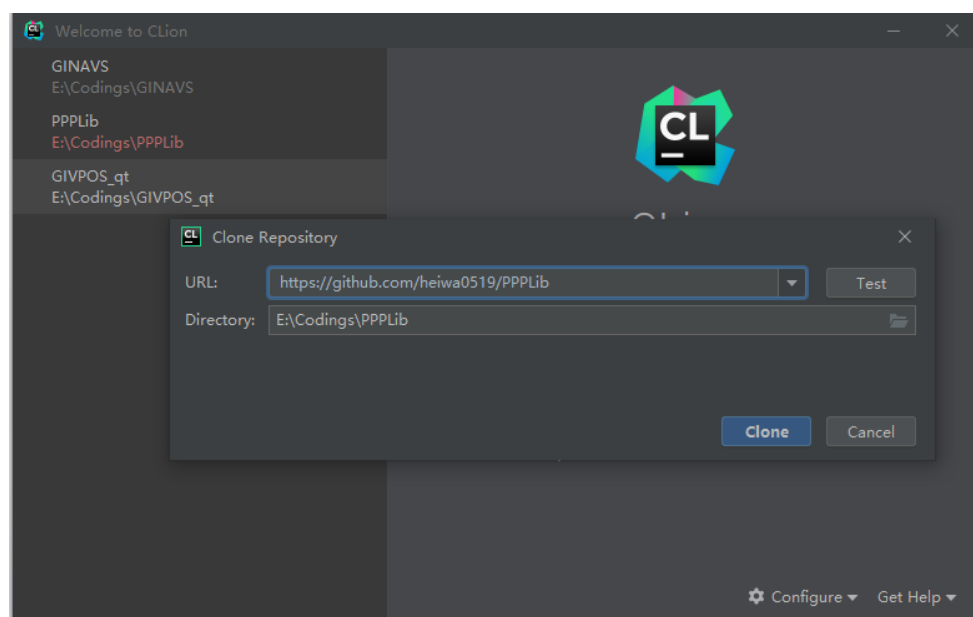


Fig. 2. Personal configuration on CLion platform under Win10

1. clone



2. cmake

```
CMake
Debug
D:\CLion\bin\cmake\win\bin\cmake.exe -DCMAKE_BUILD_TYPE=Debug -G "CodeBlocks - MinGW Makefiles" E:\Codings\PPPLib
-- The C compiler identification is GNU 6.3.0
-- The CXX compiler identification is GNU 6.3.0
-- Check for working C compiler: D:/MinGW/bin/gcc.exe
-- Check for working C compiler: D:/MinGW/bin/gcc.exe -- works
-- Detecting C compiler ABI info
-- Detecting C compiler ABI info - done
-- Detecting C compile features
-- Detecting C compile features - done
-- Check for working CXX compiler: D:/MinGW/bin/g++.exe
-- Check for working CXX compiler: D:/MinGW/bin/g++.exe -- works
-- Detecting CXX compiler ABI info
-- Detecting CXX compiler ABI info - done
-- Detecting CXX compile features
-- Detecting CXX compile features - done
-- Found PythonLibs: optimized;D:/Python3.7/python37.dll;debug;D:/Python3.7/python37_d.dll (found version "3.7.4")
-- Configuring done
-- Generating done
-- Build files have been written to: E:/Codings/PPPLib/cmake-build-debug

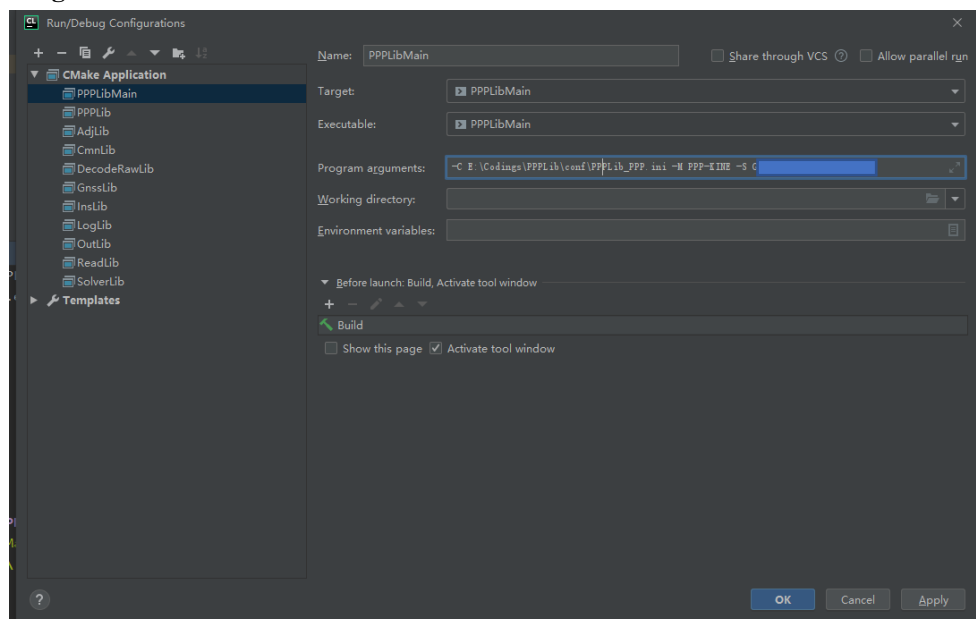
[Finished]
```

3. build

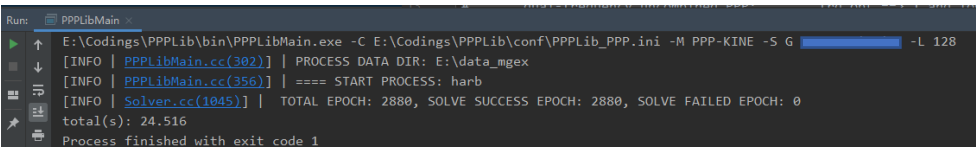
```
Messages: Build
===== [ Build | PPPLibMain | Debug ] =====
D:\CLion\bin\cmake\win\bin\cmake.exe --build E:\Codings\PPPLib\cmake-build-debug --target PPPLibMain -- -j 4
[ 12%] Built target LogLib
[ 20%] Built target CmnLib
[ 28%] Built target OutLib
[ 36%] Built target DecodeRawLib
[ 52%] Built target GnssLib
[ 60%] Built target InsLib
[ 68%] Built target AdjLib
[ 76%] Built target ReadLib
[ 84%] Built target SolverLib
[ 92%] Built target PPPLib
Scanning dependencies of target PPPLibMain
[ 96%] Building CXX object exe/CMakeFiles/PPPLibMain.dir/PPPLibMain.cc.obj
[100%] Linking CXX executable ..\..\bin\PPPLibMain.exe
[100%] Built target PPPLibMain

Build finished
```

4. configure



5. run



```
Run: PPPLibMain
E:\Codings\PPPLib\bin\PPPLibMain.exe -C E:\Codings\PPPLib\conf\PPPLib_PPP.ini -M PPP-KINE -S G -L 128
[INFO | PPPLibMain.cc(302)] | PROCESS DATA DIR: E:\data_mgex
[INFO | PPPLibMain.cc(356)] | === START PROCESS: harb
[INFO | Solver.cc(1045)] | TOTAL EPOCH: 2880, SOLVE SUCCESS EPOCH: 2880, SOLVE FAILED EPOCH: 0
total(s): 24.516
Process finished with exit code 1
```

Configuration

The configuration information of PPPLib consists of two parts, namely program arguments and processing options.

➤ Program arguments

Arguments	Mark	Value
-C	Configuration file path [essential]	String
-M	Processing mode[essential]	String SPP-KINE, PPP-KINE PPP-STATIC, PPK-KINE PPK-STATIC, IGLC-GSOF, IGLC-PPP, IGLC-PPK, IGTC- PPP, IGTC-PPK
-S	Selected GNSS[essential]	String G,B,E,R, GBJ, GR,...
-L	Debug level[essential]	Int 1 Debug 32 Warning 128 Info

➤ Processing options

The default processing options can be found [your path]/PPPLib/conf/PPPLib.ini. The explanation and value of each configuration have been noted in PPPLib.ini. Need to carefully check whether the configuration is correct before running software. For example, if you want to perform SPP-KINE, the value of **eph_opt** should set 0, otherwise, unexpected errors will occur. It is recommended to set **-L=1** for first running to debug detail information.

Dataset

PPPLib provides different dataset to evaluate its performance. This dataset contains three kinds of data, namely static GNSS data, suburban vehicle data and urban data, which can be used to evaluate the performance of PPP, PPP-AR, PPK, GNSS/INS integration mode of the software. Note some data are collected from the Internet, if you use the dataset, please cite related paper or indicate the source.

For mainland China users, please download the dataset using the [Baidu Cloud link](#) (8888).

Static GNSS data

➤ For PPP

The GNSS observations are collected from IGS-MGEX stations on December, 1, 2019, with 30s sampling intervals, including station FAA1, HARB and SGOC. The dataset includes required files to performance PPP solutions. The static/kinematic-sim, single- to multi-constellation, single- to multi-frequency PPP results can be compared and analyzed using the PPPLib software. All files can be found in folder **/data_mgex**. You can use the configuration file in folder **/conf/PPP/PPPLib_PPP_MGEX** to perform this dataset.

The software supports batch processing for PPP mode; thus, the batch processing dataset is also provided, which can be found in folder **/data_batch**. It is worth to note that the data storing in folder in according to certain rules. If you want to build your own batch folder, please follow this rule. You can use the configuration file in folder **/conf/PPP/PPPLib_PPP_BATCH** to perform this dataset. This dataset also can be used to perform PPP-AR mode.

➤ For PPK

Short-baseline GNSS data are collected from GNSS Research Centre at Curtin University (<http://gnss.curtin.edu.au/>). GNSS observations on March, 6, 2020 from station CUAA and CUBB can be used to evaluate the performance of PPK mode. All files can be found in folder **/data_cu**. You can use the configuration file in folder **/conf/PPK/PPPLib_PPK_CU** to perform this dataset.

Suburban vehicle data

This dataset is collected in a suburban environment with vehicle. The data collection platform is equipped with the GNSS (Trimble R10) and IMU (SPAN-CPT) sensors, together with accurate ground truth from SPAN-CPT system. This dataset can be used to evaluate the performance of the real-kinematic PPP, PPK and GNSS/INS integration mode. All files can be found in folder **/data_cpt0**.

Urban data

➤ From Carvig

This dataset is collected from open-source project [carvig](#). The dataset can be used to performance GNSS/INS loosely coupled and PPK/INS tightly coupled. Related files can be

converted into a unified format for easy visualization using PPPLib but missing reference truth. All files can be found in folder `/data_m39` and `/data_m39_lc`.

➤ **From UrbanNav**

This dataset is collected from open-source project [UrbanNav](https://github.com/weisongwen/UrbanNavDataset). The data information can be found at website <https://github.com/weisongwen/UrbanNavDataset>. Related files can be converted into a unified format for easy visualization using PPPLib. All files can be found in folder `/data_urban`. **In current version, this dataset is not yet supported due to its single-frequency GNSS.**

Python Scripts

PPPLib also provides some python scripts for data download, solution presented and compared, which can be found in folder `/tools`.

- **down_data.py**

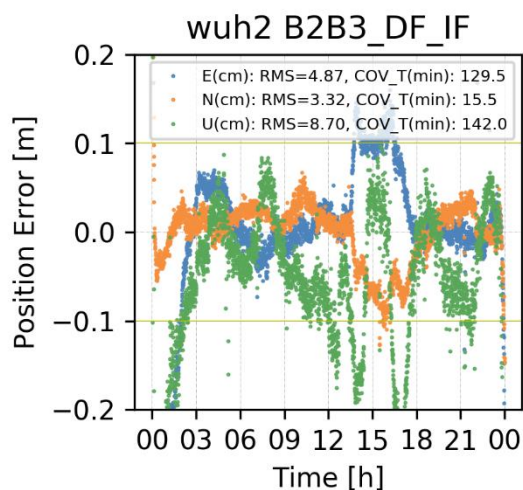
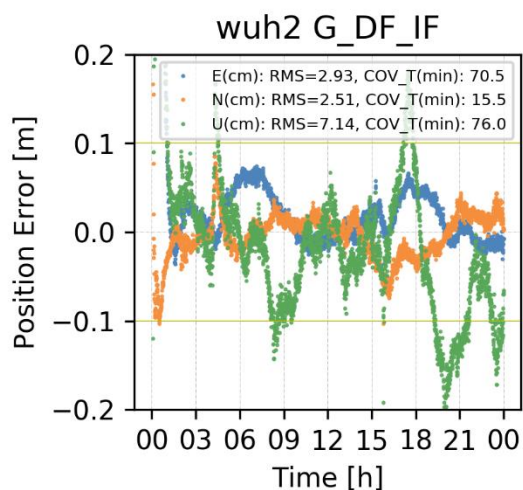
This script can be used to download observation, precise products from ftp (<ftp://igs.gnsswhu.cn>).

- **parse_sol.py**

The function of this script is to parse solution with PPPLib defined format. The position, tropospheric delay, clock and inter-system bias or satellite information can be saved in Numpy data struct.

- **plot_sol.py**

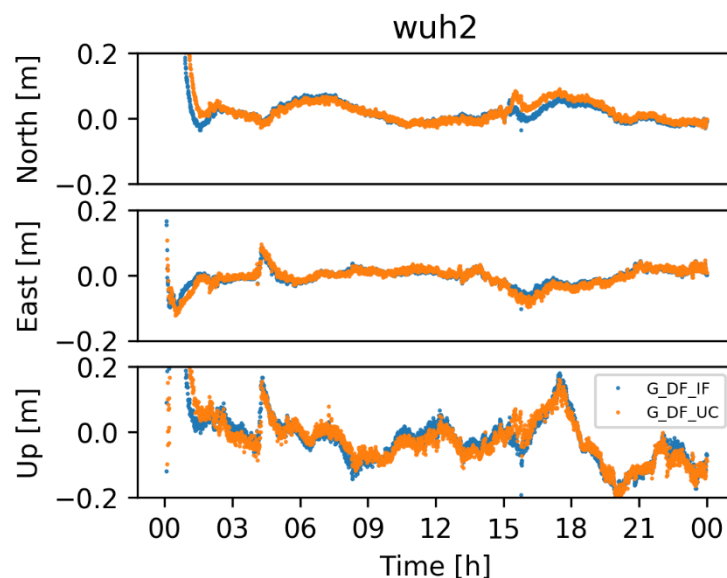
After parsing solution, you can use this script to product some figures, such as position error, tropospheric delay, ambiguity and so on.

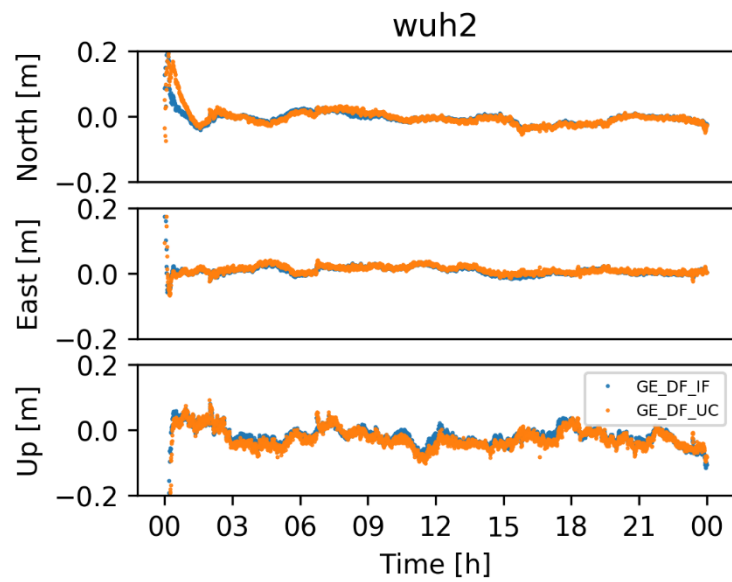
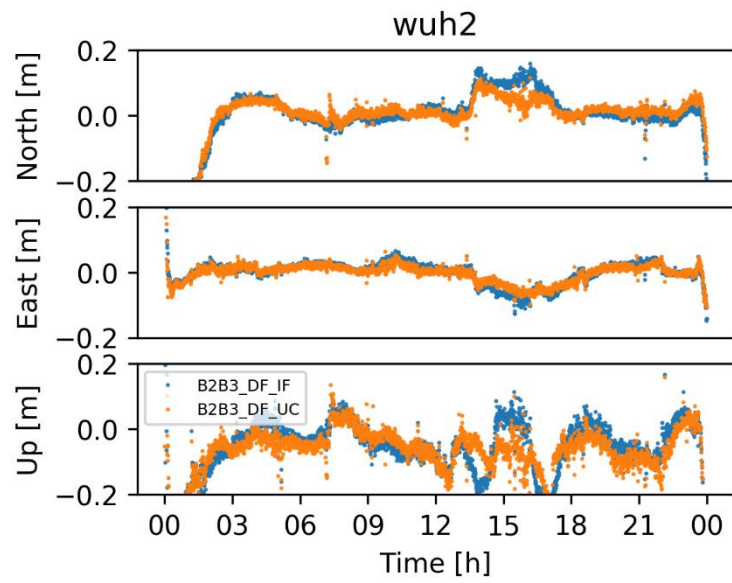


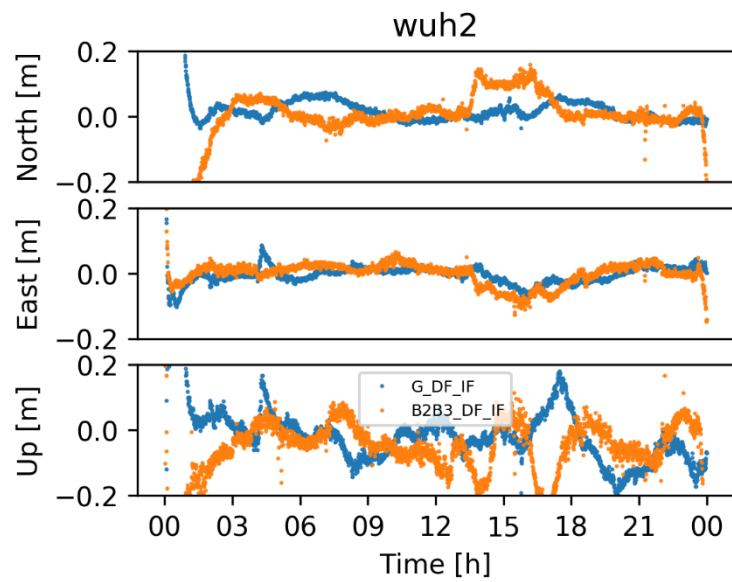
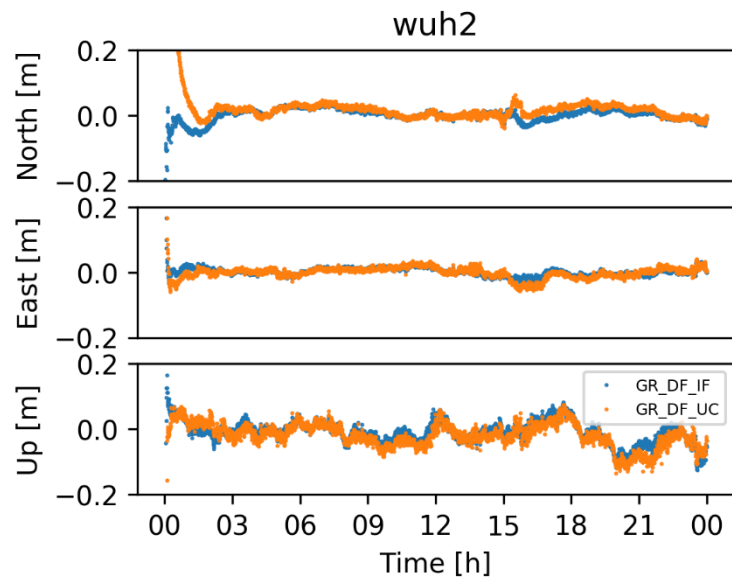
GE_DF_IF_analysis.txt									
		1.0	2.0	3.0	4.0	5.0			
1	sgoc,	1.03,	1.46,	3.53,	3.90,	19.0,	18.5,	33.0,	39.5
2	seyg,	2.37,	1.79,	6.12,	6.78,	25.0,	13.5,	30.0,	33.5
3	nya2,	1.31,	1.15,	3.35,	3.76,	20.5,	22.5,	12.0,	23.5
4	karr,	1.58,	1.10,	3.27,	3.76,	17.5,	16.5,	48.5,	50.0
5	faa1,	1.65,	1.33,	5.43,	5.46,	29.5,	15.0,	17.0,	87.0
6	tlse,	1.63,	1.31,	2.78,	3.23,	29.0,	24.5,	19.5,	45.5
7	tow2,	3.05,	2.03,	6.75,	7.67,	23.0,	10.0,	46.5,	46.0
8	urum,	1.59,	1.51,	4.58,	5.05,	27.0,	13.0,	21.0,	31.5
9	ulab,	0.78,	1.18,	3.27,	3.56,	20.0,	17.0,	21.0,	21.0
10	harb,	1.93,	1.70,	2.63,	3.67,	19.0,	10.0,	12.5,	19.0
11	kiru,	0.87,	1.16,	3.29,	3.60,	36.0,	32.0,	24.5,	38.0
12	ptgg,	1.00,	1.16,	2.06,	2.56,	13.0,	12.5,	15.5,	15.0
13	cusv,	1.11,	1.20,	3.10,	3.45,	35.0,	15.5,	23.5,	38.0
14	pado,	0.98,	1.26,	2.47,	2.92,	38.0,	14.5,	17.5,	39.0
15	iisc,	1.82,	1.40,	3.11,	3.47,	34.0,	17.0,	28.0,	76.0
16	mas1,	1.56,	1.09,	3.55,	4.00,	12.0,	13.5,	39.0,	38.5
17	sgpo,	1.13,	1.37,	2.77,	3.29,	11.0,	15.0,	19.0,	18.5
18	jfng,	1.21,	1.19,	2.79,	3.27,	14.0,	13.5,	19.0,	20.0
19	abmf,	1.66,	1.38,	3.48,	4.09,	15.5,	11.0,	15.0,	15.0
20	nium,	1.78,	0.95,	4.35,	4.81,	25.0,	14.0,	13.5,	24.5
21	nnor,	1.04,	1.03,	2.98,	3.31,	21.5,	24.5,	22.0,	27.0
22	cedu,	1.46,	1.18,	2.86,	3.42,	18.5,	19.0,	21.5,	24.0
23	djig,	0.78,	0.83,	2.12,	2.40,	13.5,	15.5,	16.5,	18.5
24	mayg,	3.45,	3.53,	7.53,	9.00,	24.5,	14.5,	31.0,	35.5

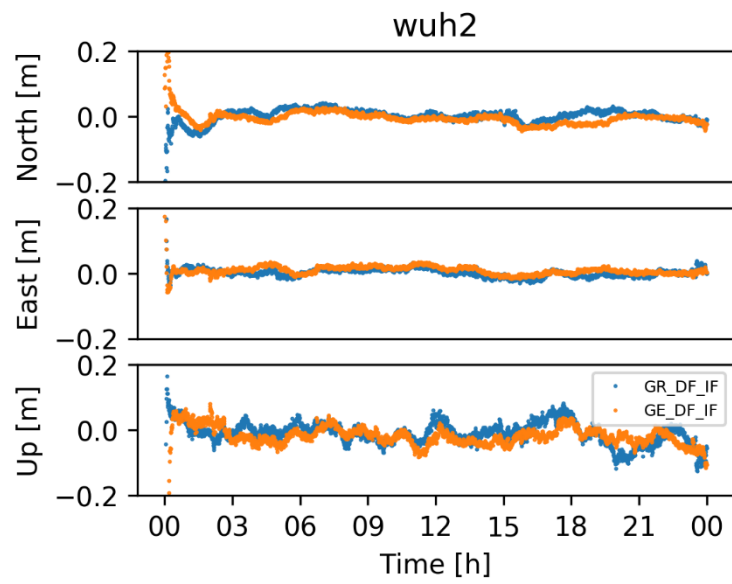
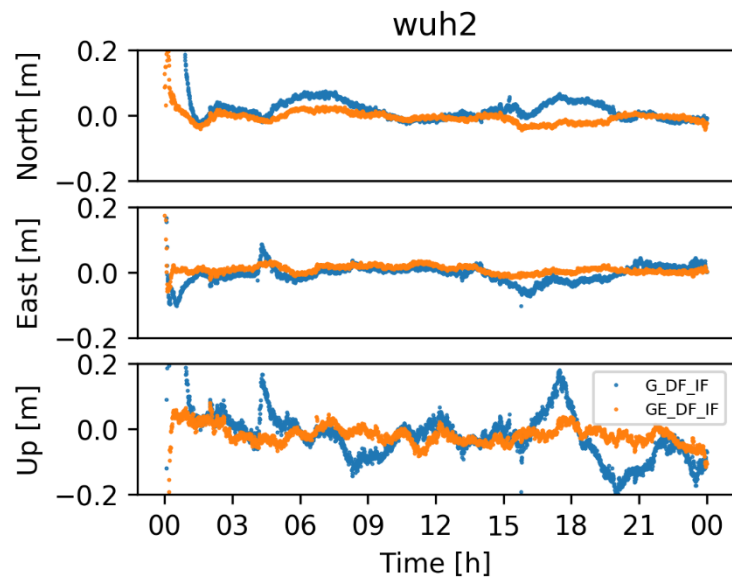
- **sols_compare.py**

The function of script compare solutions with different processing strategies and make some figures.









Appendix

PPP Models

Basic GNSS observation equations are first introduced. Then the single- to triple- frequency uncombined and ionosphere-free PPP model are derived in details.

Basic GNSS observation model

The basic undifferenced observations of original pseudorange and phase can be given as [1, 2]:

$$\begin{aligned}\tilde{P}_{r,j}^{Si} &= \rho_r^{Si} + dt_r^S - dt^{Si} + M_{w,r}^{Si} T_{w,r} + \gamma_j^S I_{r,1}^{Si} + b_{r,j}^S + b_j^{Si} + \varepsilon_{P,j}^{Si} \\ \tilde{L}_{r,j}^{Si} &= \rho_r^{Si} + dt_r^S - dt^{Si} + M_{w,r}^{Si} T_{w,r} - \gamma_j^S I_{r,1}^{Si} + N_{r,j}^{Si} + B_{r,j}^S + B_j^{Si} + \varepsilon_{L,j}^{Si}\end{aligned}\quad (1)$$

where the subscripts Si denotes the i th satellite of system S ; while the superscript r and j denote the receiver and frequency, respectively; ρ_r^{Si} is the satellite-to receiver range; $T_{w,r}^{Si}$ is slant tropospheric delay and $I_{r,1}^{Si}$ denotes the slant ionospheric delay on first frequency with $\gamma_j^S = (f_1^S / f_j^S)^2$. Here, f_j^S denotes the j th frequency. dt_r^S and dt^{Si} denote receiver and satellite clock offsets in meters, respectively; $M_{w,r}^{Si}$ denotes the wet mapping function and $T_{w,r}$ is the zenith troposphere wet delay.

$N_{r,j}^{Si}$ is the integer phase ambiguity in meters. $\varepsilon_{P,j}^{Si}$ and $\varepsilon_{L,j}^{Si}$ are the sum of measurement noise and other unmodeled error for pseudorange and phase observations. The other effects such as the relativistic effect, the phase wind-up, the phase center offset and variation, Sagnac effect, tidal loadings or hydrostatic tropospheric delay should be precisely corrected in advance following the correction models in the relevant literatures [3, 4]. Further, $b_{r,j}^S$, $B_{r,j}^S$ is the receiver uncalibrated code delay (UCD) and uncalibrated phase delay (UPD) in meters, respectively. Receiver UCD are identical for CDMA signals for all the satellites at each frequency (i.e. GPS, BDS, Galileo, QZSS), while it will be satellite-dependent and frequency-dependent for GLONASS due to the application of FDMA techniques; b_j^{Si} , B_j^{Si} is the satellite-dependent and frequency-dependent satellite UCD and UPD, respectively. For convenience, the coefficient for the ionosphere-free combination are defined as:

$$\alpha_{ij}^S = \frac{(f_i^S)^2}{(f_i^S)^2 - (f_j^S)^2}, \beta_{ij}^S = -\frac{(f_j^S)^2}{(f_i^S)^2 - (f_j^S)^2}\quad (2)$$

In general, the IGS satellite clock products are generated by using the first and second frequency ionosphere-free combination observations, thus, the satellite clocks absorb the ionosphere-free satellite UCD and the ionosphere-free satellite clock error is defined as [3]:

$$\begin{cases} dt_{12}^{Si} = dt^{Si} - b_{12}^{Si} \\ b_{12}^{Si} = \alpha_{12}^S b_1^{Si} + \beta_{12}^S b_2^{Si} \end{cases}\quad (3)$$

where dt_{12}^{Si} denotes the satellite clock correction provided by IGS satellite clock products; dt^{Si} is the real clock offsets to modeling. To make use of this ionosphere-free satellite clock product, we rewrite (3) in the form of triple-frequency:

$$\begin{cases} -dt^{Si} + b_1^S = dt_{12}^{Si} + \beta_{12}^S DCB_{12}^S \\ -dt^{Si} + b_2^S = dt_{12}^{Si} - \alpha_{12}^S DCB_{12}^S \\ -dt^{Si} + b_3^S = dt_{12}^{Si} + \beta_{12}^S DCB_{12}^S - DCB_{13}^S \end{cases} \quad (4)$$

where $DCB_{ij}^S = b_i^S - b_j^S$ is defined as the satellite differential code bias (DCB) that can be obtained from the Center for Orbit Determination in Europe (CODE) or the Multi-GNSS Experiment (MGEX). Applying the ionosphere-free satellite clocks together with DCB corrections to (1) and linearized yields:

$$\begin{aligned} p_{r,j}^{Si} &= \boldsymbol{\mu}_r^{Si} \mathbf{x}_r + dt_r^S + \mathbf{M}_{w,r}^{Si} T_{w,r} + \gamma_j^S I_{r,1}^{Si} + b_{r,j}^S + \varepsilon_{p,j}^{Si} \\ l_{r,j}^{Si} &= \boldsymbol{\mu}_r^{Si} \mathbf{x}_r + dt_r^S + \mathbf{M}_{w,r}^{Si} T_{w,r} - \gamma_j^S I_{r,1}^{Si} + N_{r,j}^{Si} + B_{r,j}^S + B_j^{Si} - b_{12}^{Si} + \varepsilon_{L,j}^{Si} \end{aligned} \quad (5)$$

where $p_{r,j}^{Si}$, $l_{r,j}^{Si}$ denote observed minus computed (OMC) values of pseudorange and phase observables, respectively; $\boldsymbol{\mu}_r^{Si}$ is the unit vector from the receiver to the satellite; \mathbf{x}_r is the vector of the receiver position increments relative to a priori position.

Single-frequency ionosphere-free PPP model

The single-frequency ionosphere-free PPP (SF-IF-PPP) model, which is originally known as group and phase ionospheric correction (GRAPHIC). The estimated parameters in SF-IF-PPP model include the receiver position, clock offsets, zenith wet tropospheric delay, ionospheric delay and float ambiguities. The SF-IF-PPP model of n-satellites and m-systems with the first frequency as an example without loss of the generality:

$$[p l_1] = \begin{bmatrix} \mathbf{A} & \mathbf{E}_C & \mathbf{M}_w & \frac{1}{2} \mathbf{E}_N \end{bmatrix} \begin{bmatrix} \mathbf{x}_r \\ \overline{dt}_r \\ T_{r,w} \\ \overline{N}_{r,1} \end{bmatrix} + [\varepsilon_{PL,1}] \quad (6)$$

with

$$\begin{cases} p l_{r,1}^{Si} = 0.5 \cdot p_{r,1}^{Si} + 0.5 \cdot l_{r,1}^{Si} \\ \overline{dt}_r^S = dt_r^S + b_{r,1}^S \\ \overline{N}_{r,1}^{Si} = N_{r,1}^{Si} + B_{r,1}^S + B_1^{Si} - b_{12}^{Si} - b_{r,1}^S \end{cases} \quad (7)$$

where \mathbf{A} is the design matrix of the coordinate vector \mathbf{x}_r . \mathbf{E}_C is a matrix of $n \times m$, of which each element is 1, corresponding to the multi-GNSS receiver clock \overline{dt}_r . \mathbf{M}_w is a n-dimensional column vector, of which each element is $\mathbf{M}_{w,r}^{Si}$, which has been defined in (1). \mathbf{E}_N is the $n \times n$ matrix

corresponding to the ambiguity $\overline{N}_{r,1}$.

Single-frequency uncombined PPP model

The estimated parameters in single-frequency uncombined PPP (SF-UC-PPP) model include the receiver position, clock offsets, zenith wet tropospheric delay, ionospheric delay and float ambiguities. Take the SF-UC-PPP model of n-satellites and m-systems with the first frequency as an example without loss of the generality:

$$\begin{bmatrix} p_1 \\ l_1 \end{bmatrix} = \begin{bmatrix} A & E_C & M_w & E_I & E_N \end{bmatrix} \begin{bmatrix} \mathbf{x}_r \\ \overline{dt}_r \\ T_{r,w} \\ \mathbf{I}_{r,1} \\ \overline{N}_{r,1} \end{bmatrix} + \begin{bmatrix} \boldsymbol{\varepsilon}_{p,1} \\ \boldsymbol{\varepsilon}_{L,1} \end{bmatrix} \quad (8)$$

with

$$\begin{cases} \overline{dt}_r^S = dt_r^S + b_{r,1}^S \\ \overline{N}_{r,1}^{Si} = N_{r,1}^{Si} + B_{r,1}^S + B_1^{Si} - b_{12}^{Si} - b_{r,1}^S \end{cases} \quad (9)$$

where $p_1 = [p_1^1, \dots, p_1^n]^T$ and $l_1 = [l_1^1, \dots, l_1^n]^T$ are the vectors of pseudorange and phase observations, respectively. E_I is $2n \times n$ matrix, the i th element for the corresponding p_1^{Si} is 1, while the i th element for l_1^{Si} is -1, corresponding to the ionospheric $\mathbf{I}_{r,1}^S$. Other symbols are defined as same in (6).

Dual-frequency ionosphere-free PPP model

The estimated parameters in dual-frequency ionosphere-free PPP (DF-IF-PPP) model include the receiver position, clock offsets, zenith wet tropospheric delay, ionospheric delay and ionosphere-free float ambiguities. Take the DF-IF PPP model of n-satellites and m-systems with the first and second frequency as an example without loss of the generality:

$$\begin{bmatrix} p_{12} \\ l_{12} \end{bmatrix} = \begin{bmatrix} A & E_C & M_w & E_N \end{bmatrix} \begin{bmatrix} \mathbf{x}_r \\ \overline{dt}_r \\ T_{r,w} \\ \overline{N}_{r,12} \end{bmatrix} + \begin{bmatrix} \boldsymbol{\varepsilon}_{p,12} \\ \boldsymbol{\varepsilon}_{L,12} \end{bmatrix} \quad (10)$$

with

$$\begin{cases} \overline{dt}_r^S = dt_r^S + b_{r,12}^S \\ \overline{N}_{r,12}^{Si} = \alpha_{12}^S N_{r,1}^{Si} + \beta_{12}^S N_{r,2}^{Si} + B_{r,12}^S + B_{12}^{Si} - b_{12}^{Si} - b_{r,12}^S \end{cases} \quad (11)$$

where $p_{12} = [p_{12}^1, \dots, p_{12}^n]^T$ and $l_{12} = [l_{12}^1, \dots, l_{12}^n]^T$ are the vectors of ionosphere-free pseudorange and phase observations with corresponding noise $\boldsymbol{\varepsilon}_{p,12}$ and $\boldsymbol{\varepsilon}_{L,12}$. $\overline{N}_{r,12}$ is ionosphere-free float

ambiguity vector. Other symbols are defined as same in (6).

Dual-frequency uncombined PPP model

The estimated parameters in dual-frequency uncombined PPP (DF-UC-PPP) model include the receiver position, clock offsets, zenith wet tropospheric delay, ionospheric delay and dual-frequency float ambiguities. Take the DF-UC-PPP model of n-satellites and m-systems with the first and second frequency as an example without loss of the generality:

$$\begin{bmatrix} \mathbf{p}_1 \\ \mathbf{l}_1 \\ \mathbf{p}_2 \\ \mathbf{l}_2 \end{bmatrix} = \begin{bmatrix} \mathbf{A} & \mathbf{E}_C & \mathbf{M}_w & \mathbf{E}_I & \mathbf{E}_{N_1} & \mathbf{E}_{N_2} \end{bmatrix} \begin{bmatrix} \mathbf{x}_r \\ \overline{dt}_r \\ T_{r,w} \\ \overline{I}_{r,1} \\ \overline{N}_{r,1} \\ \overline{N}_{r,2} \end{bmatrix} + \begin{bmatrix} \boldsymbol{\varepsilon}_{P,1} \\ \boldsymbol{\varepsilon}_{L,1} \\ \boldsymbol{\varepsilon}_{P,2} \\ \boldsymbol{\varepsilon}_{L,2} \end{bmatrix} \quad (12)$$

with

$$\begin{cases} \overline{dt}_r^S = dt_r^S + b_{r,12}^S \\ \overline{I}_{r,1}^{Si} = I_{r,1}^{Si} + \beta_{12}^S DCB_{r,12}^S \\ \overline{N}_{r,1}^{Si} = N_{r,1}^{Si} + B_{r,1}^S + B_1^{Si} - b_{12}^{Si} - b_{r,12}^S + \beta_{12}^S DCB_{r,12}^S \\ \overline{N}_{r,2}^{Si} = N_{r,2}^{Si} + B_{r,2}^S + B_2^{Si} - b_{12}^{Si} - b_{r,12}^S + \alpha_{12}^S DCB_{r,12}^S \end{cases} \quad (13)$$

where \mathbf{E}_{N_1} and \mathbf{E}_{N_2} is the matrix corresponding to the ambiguity parameters $\overline{N}_{r,1}$ and $\overline{N}_{r,2}$, respectively; the elements for the corresponding \mathbf{p} are 0, while \mathbf{l} are 1. Other symbols are defined as same in (8).

Triple-frequency ionosphere-free PPP model

Triple-frequency ionosphere-free combination can be formed as two dual-frequency ionosphere-free combinations, which have been defined in (10). Considering that the noise amplification factor, the first and second frequency ionosphere-free combinations are utilized. The estimated parameters in triple-frequency ionosphere-free PPP (TF-IF-PPP) model including the receiver position, clock offsets, receiver pseudorange inter frequency bias (IFB), zenith wet tropospheric delay, ionospheric delay and dual ionosphere-free float ambiguities. Take the TF-IF-PPP model of n-satellites and m-systems as an example without loss of the generality:

$$\begin{bmatrix} \mathbf{p}_{12} \\ \mathbf{l}_{12} \\ \mathbf{p}_{13} \\ \mathbf{l}_{13} \end{bmatrix} = \begin{bmatrix} \mathbf{A} & \mathbf{E}_C & \mathbf{E}_I & \mathbf{M}_w & \mathbf{E}_{N_{12}} & \mathbf{E}_{N_{13}} \end{bmatrix} \begin{bmatrix} \mathbf{x}_r \\ \overline{dt}_r \\ ifb_r \\ T_{r,w} \\ \overline{N}_{r,12} \\ \overline{N}_{r,13} \end{bmatrix} + \begin{bmatrix} \boldsymbol{\varepsilon}_{P,12} \\ \boldsymbol{\varepsilon}_{L,12} \\ \boldsymbol{\varepsilon}_{P,13} \\ \boldsymbol{\varepsilon}_{L,13} \end{bmatrix} \quad (14)$$

with

$$\begin{cases} \overline{dt}_r^S = dt_r^S + b_{r,12}^S \\ ifb_r^S = b_{r,13}^S - b_{r,12}^S \\ \overline{N}_{r,12}^{Si} = \alpha_{12}^S N_{r,1}^{Si} + \beta_{12}^S N_{r,2}^{Si} + B_{r,12}^S + B_{12}^{Si} - b_{12}^{Si} - b_{r,12}^S \\ \overline{N}_{r,13}^{Si} = \alpha_{13}^S N_{r,1}^{Si} + \beta_{13}^S N_{r,3}^{Si} + B_{r,13}^S + B_{13}^{Si} - b_{12}^{Si} - b_{r,12}^S \end{cases} \quad (15)$$

where E_i is matrix of $4n$ rows and m columns, corresponding to the IFB parameters ifb_r . Other symbols are defined as same in (10).

Triple-frequency uncombined PPP model

The estimated parameters in triple-frequency uncombined PPP (TF-UC-PPP) model including the receiver position, clock offsets, receiver pseudorange IFB, zenith wet tropospheric delay, ionospheric delay and triple-frequency float ambiguities. Take the TF-UC-PPP model of n -satellites and m -systems as an example without loss of the generality:

$$\begin{bmatrix} p_1 \\ l_1 \\ p_2 \\ l_2 \\ p_3 \\ l_3 \end{bmatrix} = \begin{bmatrix} A & E_C & E_i & M_w & E_l & E_{N_1} & E_{N_2} & E_{N_3} \end{bmatrix} \begin{bmatrix} x_r \\ \overline{dt}_r \\ ifb_r \\ T_{r,w} \\ \overline{I}_{r,1} \\ \overline{N}_{r,1} \\ \overline{N}_{r,2} \\ \overline{N}_{r,3} \end{bmatrix} + \begin{bmatrix} \varepsilon_{p,1} \\ \varepsilon_{L,1} \\ \varepsilon_{p,2} \\ \varepsilon_{L,2} \\ \varepsilon_{p,3} \\ \varepsilon_{L,3} \end{bmatrix} \quad (16)$$

with

$$\begin{cases} \overline{dt}_r^S = dt_r^S + b_{r,12}^S \\ ifb_r^S = (\beta_{12}^S / \beta_{13}^S) DCB_{r,12}^S - DCB_{r,12}^S \\ \overline{I}_{r,1}^{Si} = I_{r,1}^{Si} + \beta_{12}^S DCB_{r,12}^S \\ \overline{N}_{r,1}^{Si} = N_{r,1}^{Si} + B_{r,1}^S + B_1^{Si} - b_{12}^{Si} - b_{r,12}^S + \beta_{12}^S DCB_{r,12}^S \\ \overline{N}_{r,2}^{Si} = N_{r,2}^{Si} + B_{r,2}^S + B_2^{Si} - b_{12}^{Si} - b_{r,12}^S + \alpha_{12}^S DCB_{r,12}^S \\ \overline{N}_{r,3}^{Si} = N_{r,3}^{Si} + B_{r,3}^S + B_3^{Si} - b_{12}^{Si} - b_{r,12}^S + \gamma_3^S \beta_{12}^S DCB_{r,12}^S \end{cases}$$

The symbols are defined as same in (12).

Results analysis for single-frequency

With the station ABMF and HARB as an example, the static and kinematic positioning errors of SF-IF-, SF-UC-PPP models for GPS-only are shown in Fig. A1 and A2, respectively. The results are overall consistent with existing researches (Li et al. 2019). In static mode, the performances of SF-IF- and SF-UC-PPP models are comparable with centimeter accuracy, and the SF-IF-PPP model seems to converge slightly slower. In Kinematic mode, the epoch-wise coordinates were estimated as white noise, the positioning errors become much larger and noisier than those in static mode, which can be observed by

the standard deviation (STD) values shown in Fig. A1 and A2. From Fig. A2, only decimeter-level solutions can be achieved for kinematic mode and the solutions for uncombined PPP model is more concentrated. Then, we process the six-days data of all 17 stations to compare SF-IF- and SF-UC- PPP models statistically. The six-days average root-mean-square error (RMS) of positioning error and the coverage time in the east (E), north (N) and Up (U) direction are computed, shown in Table A1. Note that in SF-IF-PPP with kinematic mode, the coverage time of Up direction is unstable, the corresponding value instead of by 'N/A'. After converged, the averages positioning errors of static mode are 4.13, 3.24, 5.19 cm for SF-IF-PPP models, 4.07, 3.20, 5.04 cm for SF-UC-PPP models in east, north, up direction, respectively. In kinematic mode, the averages positioning errors are 13.4, 12.0, 15.1 cm for ionosphere-free PPP models, and 12.7, 11.8, 15.4 cm for uncombined PPP models. At this point, the discrepancies between the ionosphere-free and uncombined models are not surprising as the positioning accuracy. Regarding the convergence time, approximately 58.9, 32.1, 55.2 min for IF PPP models and 50.0, 30.8, 48.0 min for UC PPP models to achieve 30 cm or better accuracy in static mode, respectively. For kinematic mode, the averages of convergence time in east, north direction is 94.3 and 59.3 min for IF PPP models, while 50.2 and 31.0 min for UC PPP models. The UC PPP model shows slightly better convergence performance than IF PPP models, especially in up direction. Finally, there exist no significant discrepancies between the SF-IF-PPP and SF-UC-PPP with regard to positioning errors. However, the SF-IF-PPP is not easy to converge in kinematic mode. From the theoretical point of view, compared to SF-UC, the accuracy of carrier phase will be damaged by ionosphere-free operator, as a result, the pseudorange noise leads to the worse convergence performance.

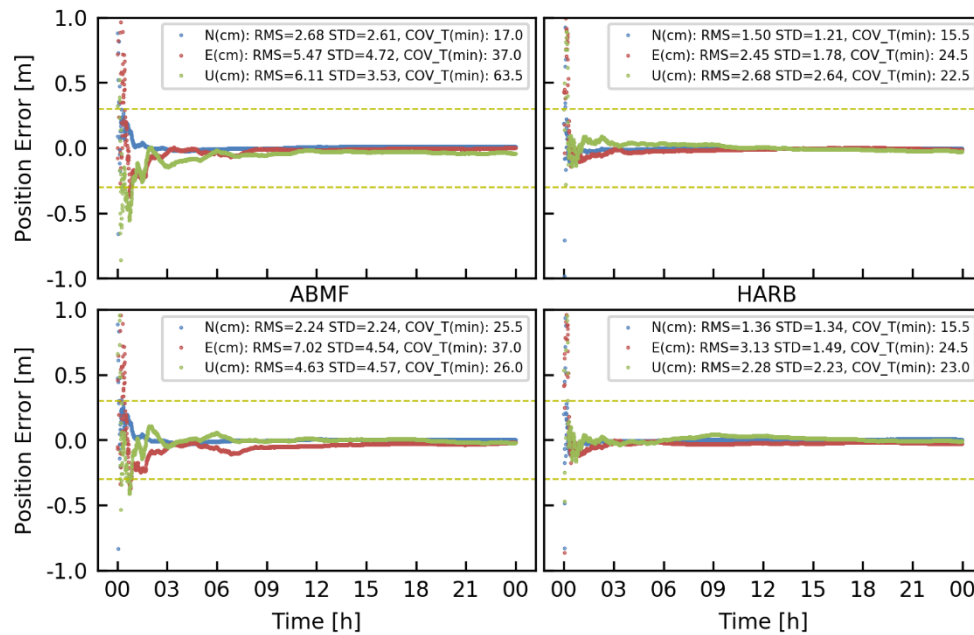


Fig. A1 Positioning errors with SF-IF- (top) and SF-UC-PPP (bottom) models for GPS-only in static mode at station AMBF (left) and HARB (right), respectively

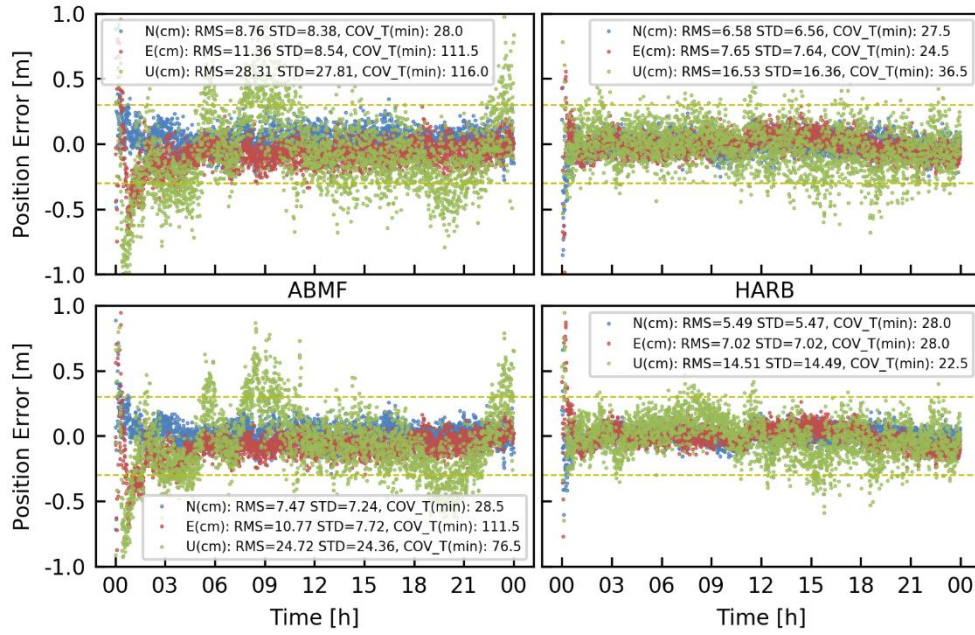


Fig. A2 Positioning errors with SF-IF- (top) and SF-UC-PPP (bottom) models for GPS-only in kinematic mode at station AMBF (left) and HARB (right), respectively

Table A1 RMS of Positioning errors (cm) and convergence time (min) with SF-IF- and SF-UC-PPP models in east, north and up direction

	Ionosphere-free		Uncombined	
	Static	Kinematic	Static	Kinematic
East	4.13 58.9	13.4 94.3	4.07 50.0	12.7 50.2
North	3.24 32.1	12.0 59.3	3.20 30.8	11.8 31.0
Up	5.19 55.2	15.1 N/A	5.40 48.0	15.4 76.0

Comparison of ionosphere-free and uncombined

The dual-frequency IF or UC model is common algorithm to perform PPP solution. In order to evaluate the discrepancies between DF-IF and DF-UC PPP models, we take G-only, G+B (GB), G+E (GE), G+R (GR) combinations with kinematic mode to perform dual-frequency PPP solution. Fig. 3 shows the DF-IF- and SF-UC-PPP averages positioning errors of all 17 stations from doy-342 to doy-347 with different GNSS system combinations scenario in east (E), north (N) and up (U) components and the average convergence time are presented in Fig. A4, corresponding statistics are shown in Table A2.

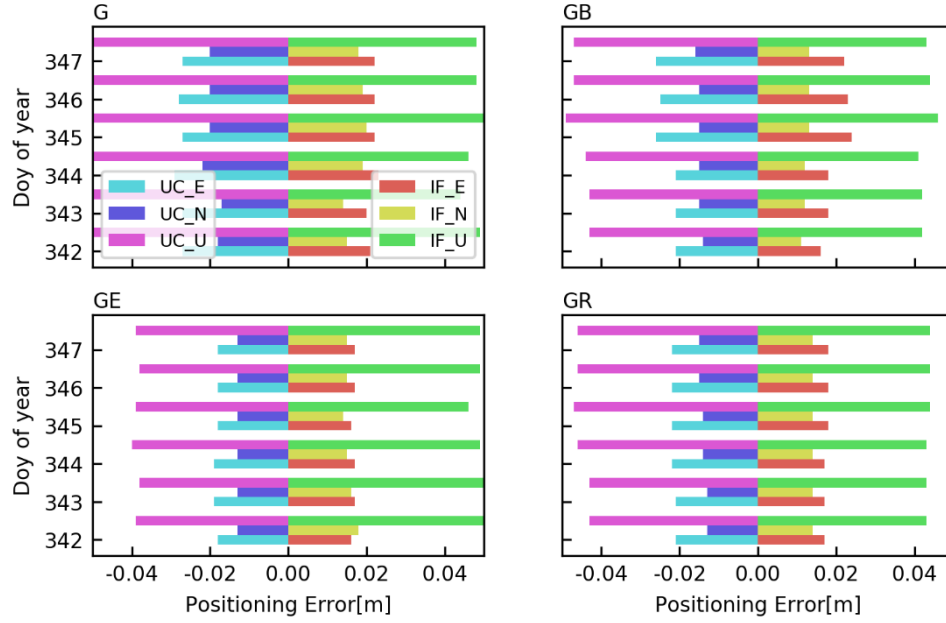


Fig. A3 Positioning errors of ionosphere-free (IF) and uncombined (UC) PPP models with different system combinations in east, north, up and 3D direction, respectively

By comparing the positioning accuracy of IF and UC PPP models from different GNSS combinations, we can find that the discrepancies between the IF and UC models are not surprising. The averages positioning errors of four schemes in E, N and U components as well as 3D direction are 1.90, 1.34, 4.60, 5.26 cm for IF and 2.27, 1.54, 4.52, 5.60 cm for UC. Another crucial performance indicator, the averages convergence time in E, N, U and 3D direction, are 25.6, 17.0, 25.7, 34.5 min for IF and 27.5, 17.8, 26.5, 39.0 min for UC, respectively. Among the four groups of solutions, the performances of IF and UC PPP models are comparable to each other for both positioning accuracy and convergence time. This is reasonable when we acknowledge the following facts. On the one hand, the equivalence properties between IF and UC models has been documented in the relevant literatures (Guo et al. 2016; Xu and Xu 2007). The solution vector and the variance-covariance matrix are identical that can be proved by algebraic linear transformations (Xu and Xu 2007). Thus, none of the models will lead to better solutions than the others (Guo et al. 2016), coinciding with the practical results. On the other head, the differences between IF and UC PPP models mainly lie in the priori information (random-walk process in this paper) is added on the ionosphere parameters, which will lead to different solutions. Moreover, initial time and re-initialization stages can also be attributed to the differences in parameterizations. Nevertheless, such small differences can be neglected in kinematic mode, but one should not use uncombined PPP models if the high-performance computing are expected.

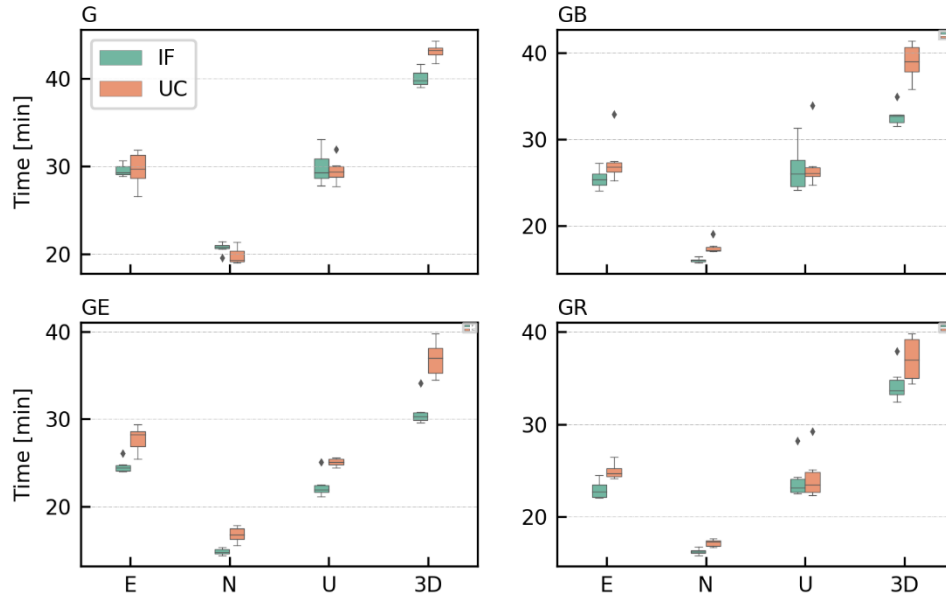


Fig. A4 Convergence time of ionosphere-free (IF) and uncombined (UC) PPP models with different GNSS system combinations in east, north, up and 3D direction, respectively

Table A2 RMS of Positioning errors (cm) and convergence time (min) using DF-IF- and DF-UC-PPP models with different GNSS system combinations in east, north, up and 3D direction, respectively.

	Ionosphere-free				Uncombined			
	East	North	Up	3D	East	North	Up	3D
G	2.17 29.5	1.18 20.7	4.75 29.9	5.58 40.3	2.75 29.7	1.95 19.8	5.13 29.5	6.20 43.1
GB	2.02 25.4	1.23 16.0	4.30 26.6	5.02 32.7	2.33 27.5	1.50 17.5	4.55 27.2	5.37 39.0
GE	1.67 24.6	1.55 14.8	5.03 22.4	5.58 30.8	1.83 27.7	1.30 16.8	3.88 25.1	4.48 36.9
GR	1.75 22.9	1.40 16.3	4.35 24.0	4.85 34.3	2.17 24.9	1.40 17.2	4.52 24.3	5.18 37.1
Average	1.90 25.6	1.34 17.0	4.60 25.7	5.26 34.5	2.27 27.5	1.54 17.8	4.52 26.5	5.60 39.0

Example of triple-frequency

As mentioned before, the IFCB parameters are not be properly handled in current version of software. The results in Pan et al. (2017) indicated that the positioning accuracy without IFCB consideration significantly degraded for GPS Block IIF satellites. However, not all constellations, including the BDS-3 satellites and Galileo, have this IFCB issues, and their marginal IFCB variations may be ignored (Pan et al. 2018). Base on it, we taken TF-UC-PPP models with BDS-only using station SGOC as example to test the triple-frequency PPP performance of the software. In our processing, the inter-system bias (ISB) between BeiDou-2 and BeiDou-3 are ignored, the three frequencies are B1I, B2I, B3I for BeiDou-2 satellites and B1I, B2a, B3I for BeiDou-3. Fig. A5 shows the number of tracked satellites, used satellites and PDOP values on December, 1, 2019. The number of tracked satellites is from 16 to 22 with PDOP values all smaller than 1.8. The positioning performances of TF-UC-PPP models with static and kinematic mode are shown in Fig. A6. RMS and STD of positioning error as well as convergence time are shown in figure. On station SGOC, BDS-only TF-UC-PPP can achieve a positioning accuracy of 1.86, 2.31, 1.30 cm and 2.65, 3.60, 3.80 cm in E, N, U direction with static and kinematic mode,

respectively. Additionally, the convergence time of BDS-only TF-UC-PPP is 24.5, 38.5, 43.5 min and 28.0, 56.5, 27.0 min in E, N, U direction with static and kinematic mode, respectively. This result indicates that PPPLib in its way to perform triple-frequency PPP. However, it I swarth mentioning that this result is special case and is not representative. Some bugs still exist in PPPLib for triple-frequency PPP that need to be further investigated.

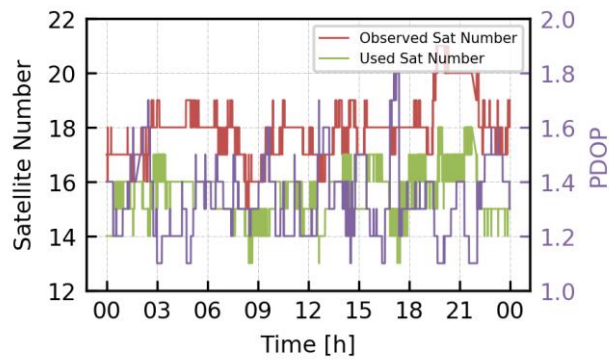


Fig. A5 Number of tracked, used satellites and the corresponding PDOP values for station SGOC on December, 1, 2019

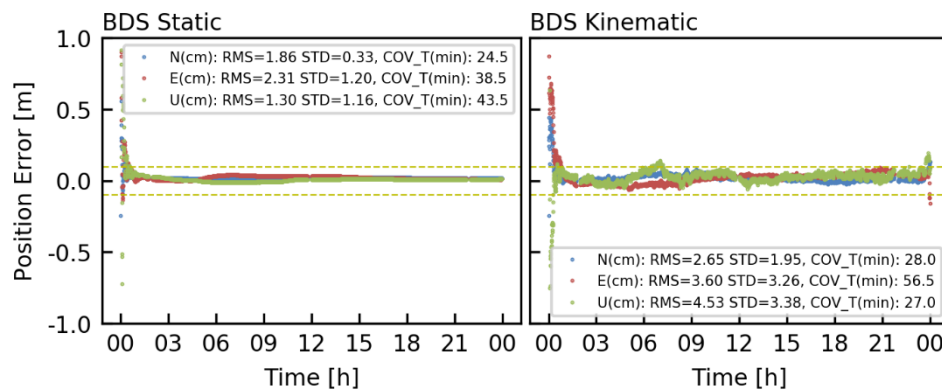


Fig. A6 Positioning errors with TF-UC-PPP model for BDS-only in static (left) and kinematic (right) mode at station SGOC, respectively

Acknowledgement

First of all, I pay tribute to Mr. Tomoji Tksu, the author of [RTKLIB](#) software. I admire him for his selfless open source spirit and elegant programming. The most functions of PPPLib are refer to [RTKLIB](#). The software is also refers to [rtkexplorer](#), HPRTK of [Mowen Li](#) from Shandong university, the [carvig](#) software of Jinlan Su from Wuhan University, the [GAMP](#) software of Feng Zhou from Shangdong University of Science and Technology, the [Multi-Sensor Fusion](#) software of Wentao Fang from Wuhan University, the [MG-APP](#) software of Gongwei Xiao from Institute of Geodesy and Geophysics, Chinese Academy of Sciences and the [PINS](#) software of Gongmin Yan from Northwestern Polytechnic University. The [easyloggingpp](#) is used to log information and [Eigen](#) is used to perform matrix operations. Thanks to the authors of above software. Many thanks are due to [Steve Hillia](#) from Notional Oceanic and Atmospheric Administration for his detailed suggestions.

Contact author

Any suggestions, corrections, and comments about PPPLib are sincerely welcomed and should be sent to:

Author: Chao Chen

QQ: 565681993

E-mail: cchen@cumt.edu.cn

Address: School of Environment and Geo-informatics, China University of Mining and Technology

References

1. Leick, A., L. Rapoport, and D. Tatarnikov, *GPS satellite surveying*. 4th edn ed. 2015: Wiley, Hoboken.
2. Zhou, F., et al., *GAMP: An open-source software of multi-GNSS precise point positioning using undifferenced and uncombined observations*. GPS Solutions, 2018. **22**(2).
3. Kouba, J., *A guide to using International GNSS Service (IGS) products*. 2009, <http://igsceb.jpl.nasa.gov/igsceb/resource/pubs/UsingIGSProductsVer21.pdf>.
4. Li, B., et al., *Single-frequency PPP models: analytical and numerical comparison*. Journal of Geodesy, 2019. **93**(12): p. 2499-2514.